## IN THE CLAIMS

This listing of the claims is provided for the convenience of the Examiner. No claims are amended, added, or cancelled in this response.

1. (Previously Presented) A method comprising:

configuring one or more processors into a D-stage processor pipeline;

transforming a sequential network application program into D-pipeline stages that collectively perform an infinite packet processing stage (PPS) loop of the sequential network application program; and

executing the D-pipeline stages in parallel within the D-stage processor pipeline to provide parallel execution of the infinite PPS loop of the sequential network application program.

2. (Original) The method of claim 1, wherein transforming the sequential application program comprises:

constructing a flow network model for the sequential application program;

selecting a plurality of preliminary pipeline stages from the flow network model; and

modifying the preliminary pipeline stages to perform control flow and variable transmission therebetween to form the D-pipeline stages.

3. (Original) The method of claim 2, wherein constructing the flow network model comprises:

transforming the application program into a static, single-assignment form;

building a control flow graph for a loop body of the application program;

building a dependence graph based on a summary graph of the control flow graph and identified, strongly-connected components (SSC) of the control flow graph; and

constructing the flow network model according to a summary graph of the dependence graph and identified SSC nodes of the dependence graph.

4. (Original) The method of claim 3, wherein constructing the flow network model comprises:

assigning a unique source node and a unique sink node to the flow network model;

adding a program node to the flow network model for each SSC node identified in the summary graph of the dependence graph;

adding a variable node to the flow network model for each variable that is defined and used by multiple program nodes;

adding a control node C to the flow network model for each SSC node identified in the summary graph of the dependence graph as a source of control dependence;

generating edges having an associated weight to connect corresponding program nodes to corresponding variable nodes;

generating edges having an associated weight to connect corresponding program nodes to corresponding control nodes; and

generating edges between the program nodes and one of the source node and the sink node.

5. (Previously Presented) The method of claim 4, wherein generating edges having an associated weight to connect corresponding program nodes to corresponding variable nodes further comprises:

(i) selecting a program node N that defines a variable node V:

(ii) adding a definition edge from node N to node V with a weight VCost to the flow network model;

(iii) repeating (i) - (ii) for each program node N that defines a variable node V;

(iv) selecting a program node M that uses a variable node W;

(v) adding an edge from the node W to the program node M with an assigned weight of infinity to the flow network model; and

(vi) repeating (iv) - (v) for each program node M that uses a variable node W.

6. (Original) The method of claim 4, wherein generating edges having an associated weight to connect corresponding program nodes to corresponding control nodes comprises:

(i) selecting a program node N that has an associated control node C;

(ii) adding a definition edge from the selected node N to the associated control node C;

(iii) associating a weight CCost to the edge;

(iv) repeating (i) - (iii) for each program node that has an associated control node;

(v) selecting a program node N having a controlled dependence on another program node M;

(vi) associating M with the control node C;

(vii) adding an edge from the associated control node C to the selected program node N;

(viii) assigning a weight of infinity to the edge; and

(ix) repeating (v) - (viii) for each node N that has a controlled dependence on another program node M.

7. (Original) The method of claim 4, wherein generating the edges between program nodes and one of the source node and the sink nodes comprises:

(i) selecting a program node without predecessor node in the flow network model;

(ii) adding an edge from the source node to the selected program node;

(iii) assigning a weight of zero to the edge;

(iv) repeating (i) - (iii) for each program node that has no predecessors;

(v) selecting a program node that has no successors in the flow network;

(vi) adding an edge from the selected program node to the sink node;

(vii) assigning a weight of zero to the added edge; and

(viii) repeating (v) - (vii) for each program node without a successor node in the flow network model.

8. (Original) The method of claim 2, wherein selecting the plurality of preliminary pipeline stages comprises:

cutting the flow network model into D-1 successive cuts, such that each cut is a balanced minimum cost cut.

9. (Original) The method of claim 8, wherein cutting is performed using an iterative balanced to push-relabel algorithm.

10. (Previously Presented) The method of claim 2, wherein modifying the preliminary pipeline stages comprises:

(a) selecting a preliminary pipeline stage;

(b) altering the selected preliminary pipeline stage to enable proper transmission of live variables and control flow to and from the selected preliminary pipeline stage; and

(c) (a) – (b) for each preliminary pipeline stage to form the D-pipeline stages of a parallel network application.

11. (Previously Presented) An article of manufacture including a machine readable medium having stored thereon instructions which may be used to program a system to perform a method, comprising:

configuring one or more processors into a D-stage processor pipeline;

transforming a sequential network application program into D-pipeline stages that collectively perform an infinite packet processing stage (PPS) loop of the sequential network application program; and

executing the D-pipeline stages in parallel within the D-stage processor pipeline to provide parallel execution of the infinite PPS loop of the sequential network application program.

12. (Previously Presented) The article of manufacture of claim 11, wherein transforming the sequential application program comprises:

constructing a flow network model for the sequential network application program;

selecting a plurality of preliminary pipeline stages from the flow network model; and

modifying the preliminary pipeline stages to perform control flow and variable transmission therebetween to form the D-pipeline stages

13. (Original) The article of manufacture of claim 12, wherein constructing the flow network model comprises:

transforming the application program into a static, single-assignment form;

building a control flow graph for a loop body of the application program;

building a dependence graph based on a summary graph of the control flow graph and identified, strongly-connected components (SSC) of the control flow graph; and

constructing the flow network model according to a summary graph of the dependence graph and identified SSC nodes of the dependence graph.

14. (Original) The article of manufacture of claim 13, constructing the flow network model comprises:

assigning a unique source node and a unique sink node to the flow network model;

adding a program node to the flow network model for each SSC node identified in the summary graph of the dependence graph;

adding a variable node to the flow network model for each variable that is defined and used by multiple program nodes;

adding a control node C to the flow network model for each SSC node identified in the summary graph of the dependence graph as a source of control dependence;

generating edges having an associated weight to connect corresponding program nodes to corresponding variable nodes;

generating edges having an associated weight to connect corresponding program nodes to corresponding control nodes; and

generating edges between the program nodes and one of the source node and the sink node.

15. (Previously Presented) The article of manufacture of claim 14, generating edges having an associated weight to connect corresponding program nodes to corresponding variable nodes further comprises:

(i) selecting a program node N that defines a variable node V;

(ii) adding a definition edge from node N to node V with a weight VCost to the flow network model;

(iii) repeating (i) - (ii) for each program node N that defines a variable node V;

(iv) selecting a program node M that uses a variable node W;

(v) adding an edge from the node W to the program node M with an assigned weight of infinity to the flow network model; and

(vi) repeating (iv) - (v) for each program node M that uses a variable node W.

16. (Original) The article of manufacture of claim 14, wherein generating edges having an associated weight to connect corresponding program nodes to corresponding control nodes comprises:

(i) selecting a program node N that has an associated control node C;

(ii) adding a definition edge from the selected node N to the associated control node C;

(iii) associating a weight CCost to the edge;

(iv) repeating (i) - (iii) for each program node that has an associated control node;

(v) selecting a program node N having a controlled dependence on another program node M;

(vi) associating M with the control node C;

(vii) adding an edge from the associated control node C to the selected program node N;

(viii) assigning a weight of infinity to the edge; and

(ix) repeating (v) - (viii) for each node N that has a controlled dependence on another program node M.

17. (Original) The article of manufacture of claim 14, wherein generating the edges between program nodes and one of the source node and the sink nodes comprises:

(i) selecting a program node without predecessor node in the flow network model;

(ii) adding an edge from the source node to the selected program node;

(iii) assigning a weight of zero to the edge;

(iv) repeating (i) - (iii) for each program node that has no predecessors;

(v) selecting a program node that has no successors in the flow network;

(vi) adding an edge from the selected program node to the sink node;

(vii) assigning a weight of zero to the added edge; and

(viii) repeating (v) - (vii) for each program node without a successor node in the flow network model.

18. (Original) The article of manufacture of claim 12, wherein selecting the plurality of preliminary pipeline stages comprises:

cutting the flow network model into D-1 successive cuts, such that each cut is a balanced minimum cost cut.

19. (Original) The article of manufacture of claim 18, wherein cutting is performed using an iterative balanced to push-relabel algorithm.

20. (Original) The article of manufacture of claim 12, wherein modifying the preliminary pipeline stages comprises:

selecting a preliminary pipeline stage;

altering the selected preliminary pipeline stage to enable proper transmission of live variables to and from the selected preliminary pipeline stage;

altering the selected preliminary pipeline stage to enable proper transmission of control flow to and from the selected preliminary pipeline stage; and

repeating the selecting, altering and altering for each preliminary stage to form the D-pipeline stages of a parallel network application.

21. (Previously Presented) A method comprising:

constructing a flow network model from a sequential network application program;

cutting the flow network model into a plurality of preliminary pipeline stages; and

transforming the preliminary pipeline stages to perform control flow and variable transmission therebetween to form D-pipeline stages that collectively perform an infinite packet processing stage (PPS) loop of the sequential network application program to enable parallel execution of the infinite PPS loop of the sequential network application program.

22. (Previously Presented) The method of claim 21, wherein transforming the preliminary application program stages comprises:

(i) electing a preliminary application program stage;

(ii) selecting a control flow graph generated for the infinite PPS loop corresponding to the selected preliminary application program stage;

(iii) removing instructions from the control flow graph if the instruction is not contained within the selected preliminary pipeline stage;

(iv) transforming the selected control flow graph according to variables and control objects transmitted from a prior stage;

(v) reconstructing the PPS loop from the transformed control flow graph to form a pipeline stage; and

repeating (i) - (v) for each preliminary pipeline stage to form D-pipeline stages of a parallel network application program.

23. (Original) The method of claim 22, wherein transforming the control flow further comprises:

selecting values for control objects transmitted from a prior pipeline stage on entry to the control flow graph;

for each control object received from the prior pipeline stage, constructing a conditional instruction using the control object; and

replacing corresponding conditional nodes within the CFG with the conditional instruction.

24. (Original) The method of claim 22, wherein transforming the control flow further comprises:

selecting values for variables that are transmitted from a prior pipeline stage; and

for each variable transmitted to a next pipeline stage, setting a value of the variable to a distinctive temporary following definition of the variable within the control flow graph.

25. (Original) The method of claim 22, wherein transforming the control flow graph further comprises:

for each control object to be transmitted to a next pipeline stage, placing an alternate value of the control object in each alternate successor node of a conditional node associated with the control object in the control flow graph; and

transmitting live set data to a next pipeline stage at exit of the control flow graph.

26. (Previously Presented) An article of manufacture including a machine readable medium having stored thereon instructions which may be used to program a system to perform a method, comprising:

constructing a flow network model from a sequential network application program;

cutting the flow network model into a plurality of preliminary pipeline stages; and

transforming the preliminary pipeline stages to perform control flow and variable transmission therebetween in order to form D-pipeline stages that collectively perform an infinite packet processing stage (PPS) loop of the sequential network application program to enable parallel execution of the infinite PPS loop of the sequential network application program.

27. (Original) The article of manufacture of claim 26, wherein transforming the preliminary application program comprises:

(i) electing a preliminary application program stage;

(ii) selecting a control flow graph generated for a packet processing stage (PPS) loop corresponding to the selected preliminary application program stage;

(iii) removing instructions from the control flow graph if the instruction is not contained within the selected preliminary pipeline stage;

(iv) transforming the selected control flow graph according to variables and control objects transmitted from a prior stage;

(v) reconstructing the PPS loop from the transformed control flow graph to form a pipeline stage; and

repeating (i) - (v) for each preliminary pipeline stage to form D-pipeline stages of a parallel network application program.

28. (Original) The article of manufacture of claim 26, wherein transforming the control flow graph further comprises:

selecting values for control objects transmitted from a prior pipeline stage on entry to the control flow graph;

for each control object received from the prior pipeline stage, constructing a conditional instruction using the control object; and

replacing corresponding conditional nodes within the control flow graph with the conditional instruction.

29. (Original) The article of manufacture of claim 26, wherein transforming the control flow graph further comprises:

selecting values for variables that are transmitted from a prior pipeline stage; and

for each variable transmitted to a next pipeline stage, setting a value of the variable to a distinctive temporary following definition of the variable within the control flow graph.

30. (Original) The article of manufacture of claim 28, wherein transforming the control flow graph further comprises:

for each control object to be transmitted to a next pipeline stage, placing an alternate value of the control object in each alternate successor node of a conditional node associated with the control object in the control flow graph; and

transmitting live set data to a next pipeline stage at exit of the control flow graph.

31. (Previously Presented) An apparatus, comprising:

a processor;

a memory coupled to the processor, the memory including a compiler to cause transformation of a sequential network application program into D-pipeline stages that collectively perform an infinite packet processing stage (PPS) loop of the sequential network application program to enable parallel execution of the D-pipeline stages within a D-stage processor pipeline to provide parallel execution of the infinite PPS loop of the sequential network application program.

32. (Original) The apparatus of claim 31, wherein the compiler to cause construction of a flow network model for the sequential application program, to cause selection of a plurality of

preliminary pipeline stages from the flow network model and to cause modification of the preliminary pipeline stages to perform control flow and variable transformation therebetween to form the D-pipeline stages.

33. (Original) The apparatus of claim 32, wherein the compiler to cause D-1 successive cuts of the flow network mode, such that each cut is a balanced, minimum cost cut to form the D-preliminary pipeline stages.

34. (Previously Presented) A system comprising:

a processor;

a memory controller coupled to the processor; and

a DDR SRAM memory coupled to the processor, the memory including a compiler to cause transformation of a sequential network application program into D-application program stages that collectively perform an infinite packet processing stage (PPS) loop of the sequential network application program to enable parallel execution of the D-application program stages within a D-stage processor pipeline to provide parallel execution of the infinite PPS loop of the sequential network application program.

35. (Original) The system of claim 34, wherein the compiler to cause construction of a flow network model for the sequential application program, to cause selection of a plurality of preliminary pipeline stages from the flow network model and to cause modification of the preliminary pipeline stages to perform control flow and variable transformation therebetween to form the D-pipeline stages.

36. (Original) The system of claim 35, wherein the compiler to cause D-1 successive cuts of the flow network mode, such that each cut is a balanced, minimum cost cut to form the D-preliminary pipeline stages.